Analytical Queries on Road Networks: An Experimental Evaluation of Two System Architectures

Shangfu Peng & Hanan Samet, SIGSPATIAL 2015

Payas Rajan • UC Riverside • 10.06.2016
Outline

Background
- Spatial Analytical Queries

Architectures compared
- Hybrid architecture
- Integrated architecture

Evaluation
- Throughput
- Effect of density

Conclusions
Outline

Background
- Spatial Analytical Queries

Architectures compared
- Hybrid architecture
- Integrated architecture

Evaluation
- Throughput
- Effect of density

Conclusions
Spatial Analytical Queries

Case 1

- I have a dataset of taxi trips such that each trip has a latitude and longitude values for both a pickup and a drop off point, as well as the trajectory information for each taxi trip.
- I want to obtain the total number of miles travelled by each taxi this month.

Case 2

- I have the geocoded address of my patients and the locations of my clinics. Our hospital has more than 500 clinics across the country. Each patient is assigned to the nearest clinic.
- I want to get the average drive time for patients per clinic.
Another example

- Given a dataset containing the residence and office locations of all the people in a city.
- Find the number of jobs within a 10km driving radius.

Image source: Author’s slides
Spatial Analytical Queries

- Require computing shortest ‘network’ distances rather than ‘euclidean’ distances. Why?

- Typically require *hundreds of thousands* to *several millions* of network distance computations per second. Why?
  - Typically used for generating insights into the data in the form of reports or visual representation.
  - The queries may join two or more datasets on the basis of the network distance to other objects on the road network.

- An ‘umbrella term’ for large-scale KNN, CNN and distance matrix queries.
  - Have two distinct access patterns: ‘one-to-one’ and ‘one-to-many’ or ‘many-to-many’.

- Commercial solutions:
  - Google Maps API allows limited ‘distance matrix’ computations on road networks.
  - Other solutions just use Euclidean distances.
What’s the hype about network distances?

5. Flat Top
- 177 reviews
- American (New), Cafes
- 1.3 Miles

6. Santouka Ramen
- 412 reviews
- Ramen
- 0.6 Miles

7. Daylinin Yeri
- 130 reviews
- Turkish
- 1.0 Miles

8. Giulia’s Kitchen
- 72 reviews
- American (New), Burgers, American (Traditional)
- 0.6 Miles
## Access patterns for Analytical Queries

Analytical queries perform following access patterns on the road network:

- **One-to-one**: A set of source-target pairs
- **One-to-many**: $k$ Nearest Neighbours
- **Many-to-Many**: Distance Matrix

The following table summarizes the access patterns and the corresponding algorithms:

<table>
<thead>
<tr>
<th>Algorithm Type</th>
<th>Access Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>One-to-One</strong></td>
<td></td>
</tr>
<tr>
<td>Scan-based (in memory)</td>
<td>Contraction Hierarchies, TNR, etc</td>
</tr>
<tr>
<td>Lookup-based (in database)</td>
<td>-Distance Oracle, Hub labeling</td>
</tr>
<tr>
<td><strong>One-to-Many</strong></td>
<td></td>
</tr>
<tr>
<td>Scan-based (in memory)</td>
<td>Dijkstra’s algorithm, etc</td>
</tr>
</tbody>
</table>

Table source: Author’s slides
Outline

Background
- Spatial Analytical Queries

Architectures compared
- Hybrid architecture
- Integrated architecture

Evaluation
- Throughput
- Effect of density

Conclusions
Architectures to solve analytical queries

**Hybrid architecture**

- Has two parts:
  - Uses a database to store and query the spatial datasets (Points of Interest, relations and attributes).
  - Has a separate module for loading the road network in main memory and computing shortest distances.

- Eg: ArcGIS Network Analyst.

**Integrated architecture**

- Incorporates the road network inside the database as a single relation by storing it as an $\varepsilon$-Distance Oracle [Sankaranarayanan & Samet, ICDE ’09].

- Allows KNN and trajectory queries to be written using slightly modified SQL.
Hybrid architecture

- The analysis tool partitions the query processing into two parts:
  - Retrieve the objects from the spatial database
  - Compute the distance between them using the road-network module.
- Road network scanner implements the access patterns on the road networks. The following operators are defined:
  - SCAN()
  - SCAN_UNTIL_K()
  - SCAN_UNTIL_DIST()
Approximate Distance Oracles [Thorup & Zwick, Journal of ACM ‘05]

- Let $G(V,E)$ be a graph. We seek to answer the query “What is the network distance between nodes $s$ and $t$?”
- Simplest solution: Run the Dijkstra’s algorithm
  - [Run time: $O(m+\log n)$]
  - [Extra space: None]
- However, this is slow for a lot of applications. Now, what if we’re allowed to do some preprocessing on the graph and somehow store that information?
- A data structure for storing approximate network distances in a graph. -- Distance Oracle.
Given a well-separated pair decomposition of a road network, this version of the DO stores the $\epsilon$-approximate distance between nodes.

$$\forall s \in A, t \in B, (1 - \epsilon)d_\epsilon(A, B) \leq d_G(s, t) \leq (1 + \epsilon)d_\epsilon(A, B)$$
Integrated Architecture

- Embed the Distance Oracle into the architecture and index it with a B-Tree.
- “The precomputation process decomposed the road network with $n$ vertices into $O(n^2)$ triples $(A, B, d)$ stored in a relational table, such that $A$ and $B$ are denoted by blocks in a PR quadtree and $d$ is the network distance that approximates the network distance between every pair of vertices contained in $A$ and $B$.”
- A Distance query then becomes a simple B-Tree lookup.
- The biggest leverage is the use (or re-use) of SQL. eg:

```sql
-- Road distance between White House and US Capitol
SELECT DIST(38.8977, -77.0366, 38.8898, -77.0091)
-- This produces 2144.7 (meters)
```
Another example of a ‘complex’ operation

- Suppose that we have:
  - a relation `houses` (id, lat, lon) corresponding to the location of all houses available for sale
  - another relation `parks` (id, lat, lon) corresponding to the location of all parks.
- We want to find up to 100 houses with the maximum number of parks that lie within 0.5 km of road distance from the houses sorted by the number of such parks.

```sql
SELECT id, count(*) as count
FROM ( SELECT houses.id as id,
            DIST(houses.lat, houses.lon, 
                 parks.lat, parks.lon) as dist
            FROM houses, parks
        ) as foo
WHERE dist < 500 -- 0.5km in meters
GROUP BY id
ORDER BY count DESC
LIMIT 100;
```
Architectures to solve analytical queries

Hybrid architecture

- APPLICATIONS
- ANALYSIS TOOL
  - SQL
  - ROAD NETWORK SCANNER
    - SCANS
      - One-to-Many
      - One-to-One
  - Optimizer
- DATABASE
  - Datasets

Integrated architecture

- APPLICATIONS
  - SQL
  - Query Language Parsing
    - Logical Layer
      - DIST() Logical Layer
      - UDFs
    - Query Optimization
    - Relational Operators
  - Physical Layer
    - GIST Indices
    - Lookups
      - One-to-Many
      - One-to-One
    - DISTANCE ORACLE
  - DATASETS
Outline

Background
- Spatial Analytical Queries

Architectures compared
- Hybrid architecture
- Integrated architecture

Evaluation
- Throughput
- Effect of Density

Conclusions
Evaluation

Hybrid architecture

- Uses an in-memory implementation that stores the datasets.
- Runs on Amazon EC2 m3.2xlarge with 8vCPU and 30GB memory.
- Road network dataset from 9th DIMACS Implementation challenge with ~23M vertices and ~59M edges.
- Trajectory dataset from the SF Yellow Cabs collected by CRAWDAD containing 11,220,058 GPS entries for 537 taxis

Integrated architecture

- Runs on Amazon EC2 m3.2xlarge with 8vCPU and 30GB memory.
Evaluation: Queries

- **Throughput**

  *Question:* How many distance computations per second?

  ```sql
  SELECT x.id, y.id,
          dist(x.lat, x.lon, y.lat, y.lon) as dist
  FROM University x, Restaurant y
  ```

<table>
<thead>
<tr>
<th>Query</th>
<th>Metric</th>
<th>Integrated</th>
<th>Hybrid - 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>10K random pairs</td>
<td>Time</td>
<td>0.327 sec</td>
<td>2026 sec</td>
</tr>
<tr>
<td></td>
<td>Throughput</td>
<td><strong>30581</strong> dist/sec</td>
<td><strong>4.9</strong> dist/sec</td>
</tr>
<tr>
<td>Distance Matrix (5,96449,573)</td>
<td>Time</td>
<td>8853.9 sec</td>
<td>20139 sec</td>
</tr>
<tr>
<td></td>
<td>Throughput</td>
<td><strong>33392</strong> dist/sec</td>
<td><strong>14680</strong> dist/sec</td>
</tr>
</tbody>
</table>
Evaluation: Queries

- **Effect of density on the Distance Oracle**
  Density is defined as the ratio of the number of destinations found to the number of vertices of the road network visited by HY during its scan around each source point for a given region.
  - Time cost increases with density in the algorithms that are good for one-to-one (Integrated)
  - Time cost is not related to density in the algorithms that are good for one-to-many (Hybrid)
Conclusions
Conclusions

**One-to-one access pattern**
Integrated architecture is better than the hybrid architecture.

**One-to-many access pattern**
It depends on the density of the dataset.

**Expressive power**
SQL offered by the Integrated architecture beats Hybrid architecture.
Future work ideas

1. Can the architectures support dynamically changing weights?

2. We’re assuming the entire data is centralised. Can the DO be decentralised?

3. Can other graph-labeling methods match these architectures?
Thank You! :)